Driver Activity Monitoring Using MobileNets

Garima Tripathi, Deval Srivastava, Priyank Shah, and Saim Shaikh

Abstract This paper presents a method to monitor the driver's activity and continuously look for red flags such as distracted driving, overuse of mobile phones while driving, drowsiness, sleeping. This is achieved by using a camera-based system and the MobileNet neural network which has been fine tuned on our self-made dataset.

Keywords Computer vision · Machine learning · Activity monitoring · MobileNets · Image processing · Raspberry pi · Neural networks · Deep learning

1 Introduction

In our current world as driving technology continues to grow the driving effort required decreases. Hence, our drivers become more and more careless resulting in loss of life in many circumstances. The proposed system aims to solve this problem by developing a system to monitor driver's activity and warn them whenever necessary. The method involves deploying a neural network trained on various categories such as talking or texting on the phone, talking to co passengers, operating the radio and drinking water. In our paper we have extensively tested performance of different neural networks [1] such as Resnet-50 [2], Inception [3] and MobileNets [4]. Throughout the development our focus has been to make a system which replicates the driver's real life conditions. Hence, our network will receive the images from an IR camera allowing our system to perform during night time. Our system can be easily fitted to any existing vehicle very easily and will be intuitive to use. Our system has been developed such that it can work even in regions having extremely poor internet connectivity. The system will also be equipped with sensors to detect rash driving and will consist of security features such as fencing, fingerprint authentication to prevent thieving of the vehicle.

From the previous work and research done in this domain it can be concluded that the most popular computer vision methods include detecting driver inattention

G. Tripathi (✉) · D. Srivastava · P. Shah · S. Shaikh
Fr. Conceicao Rodrigues College of Engineering, Mumbai, India
e-mail: garima@frcrce.ac.in

V. E. Balas et al. (eds.), *Intelligent Computing and Networking*,
Lecture Notes in Networks and Systems 146,
https://doi.org/10.1007/978-981-15-7421-4_15

using head pose, eye gaze estimation or simply checking eye closure rate as well as measures such as EEG, electrocardiogram, etc. We will discuss these methods and other techniques that have been used in the next section.

The paper has been organized in the following manner, the next section will discuss related work in the given domain in depth after which we will present our method of approaching this condition and finally we will discuss all the results we have obtained from the proposed system.

2 Literature Survey

According to the research conducted by us it can be concluded that the most popular methods to solve this problem involve either driver biological measures, driver physical measures, driving performance measures or some kind of a hybrid measure [5].

Driver biological measures include biological signals like EEG, electrocardiogram (ECG), electro-oculography (EOG). These signals are collected through electrodes in contact with the skin and then analysed for fatigue and drowsiness. Physical measures involve eye closure detection and blink frequency, face position, driver gaze to detect inattention.

Driver performance measures involve various measures such as steering angle and other driving criteria. Most research that has been done in the related field has been focused on detecting driver inattention using eye gaze tracking and head pose estimation. These methods rely only on head and eye movement to detect inattention whereas in real life a driver can be distracted doing various tasks that cannot be detected by head movement alone. It has been observed that current driver monitoring systems employ statistical machine learning methods to detect driver distraction and work on a limited dataset. Some research has been done on applying deep learning technologies to solve this problem but such systems cannot be deployed in a vehicle in a cost effective manner nor do they work in night time conditions [5, 6].

3 Algorithm

For the development of our system we made use of the MobileNet Algorithm [7]. MobileNet [7] is a neural network that was developed by Google to perform on low powered devices lacking graphical GPUs that are known to accelerate neural network performance. MobileNets are small, low-latency, low-power models parameterized to meet the resource constraints of a variety of use cases, one of those use cases is that it can also be deployed on a Raspberry pi which we intend to do.

A standard convolution, filters and combines inputs into a new set of outputs in one step, but in a case of MobileNets it first uses depth wise convolution [7] that applies a single filter to each input channel. The point wise convolution then applies a 1 $\times$

1 convolution to combine the outputs the depth wise convolution. The depth wise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size, this modification allows MobileNet to be faster than its other counterparts.

For our application we have employed a MobileNet v2 [7], its the second iteration of MobileNets and now along with the depth wise separable blocks it also uses bottleneck residual layers and also adds a 1×1 expansion block whose purpose is to expand the number of channels in the data before it goes to the next block. In the proposed system we have used MobileNet v2 as it's much better than its older version, mobilenet was trained on a self made dataset of driver's performing distracted activities.

4 Proposed method

4.1 Implementation

Figure 1 describes the entire workflow of the project right from the hardware setup to the user interface. The Pi camera is mounted on an appropriate position in the dashboard of the vehicle. It is then connected to the camera port on the Raspberry Pi. The SM808 GSM + GPS module is connected to the Raspberry Pi via USB TO RS232 serial port. The GPS antenna is connected to the module and placed outside

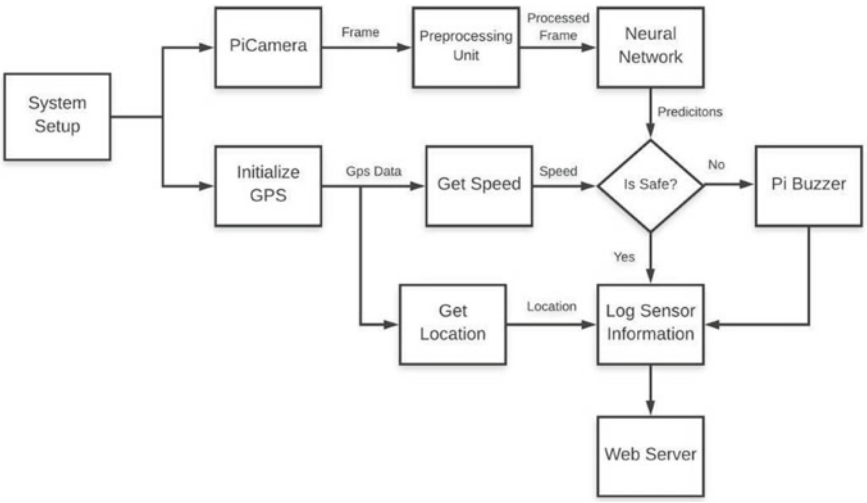

Fig. 1 Implementation flowchart

the vehicle with a clear view of the sky. The Raspberry Pi is then connected to a portable power supply via the micro USB port.

The Pi camera records footage of the driver and sends the frames to the Preprocessing Unit. The footage is recorded at a resolution of 640 * 480 at 25 frames per second. The preprocessing unit then performs basic image processing, noise reduction on each frame and resizes them to 224 * 224 * 3 then it is forwarded to the neural network which predicts the class of that image and depending on that result we declare the driver as distracted or not. If the driver is distracted the buzzer is rung to alert the driver.

For night time footage the Pi Camera is aided by 2 IR bulbs which help provide clear frames even in pitch black conditions. The frames received from the Pi Camera are processed. Alongside this we also calculate the speed of the vehicle. Once both of these operations are complete data is pushed to the administrative web server.

4.2 Dataset and Training

When we were looking for a dataset that would be able to suffice our needs for the classification tasks we came across many publicly available datasets one of them being the NTHU Driver Drowsiness Detection Dataset [8], Kaggle Statefarm Dataset [9] but none of them proved to be satisfactory as our trained neural networks were not performing as expected in real world conditions [10]. After this realization we started working on creating a real world dataset curated to our task. We recorded 6 drivers performing various distracted activities across multiple cars. We recorded drivers performing activities such as talking on the phone, texting while driving, drinking, sleeping, yawning. All of these activities were performed in simulated environments where the drivers are not actually driving. The Dataset was recorded using a pi camera as it's the camera that will be feeding images to the neural network. We recorded images for the night dataset by using an IR camera and IR lights. The database statistics are presented in Table 1.

Figures 2 and 3 we can see the dataset samples from night and from day. The images are in order of Sleeping, Talking on the phone, Drinking, Texting, Yawning.

Table 1 Database statistics

S. No.	Class	Count (day + night)
1	Safe driving	5000
2	Talking on phone	5000
3	Texting on phone	5000
4	Drinking	5000
5	Sleeping	5000
6	Yawning	5000

Fig. 2 Dataset samples taken during day

Fig. 3 Dataset samples taken during night

This section will discuss how we trained our model on the above dataset. We have used transfer learning to train our neural network as we have used the model of MobileNet v2 and only trained the last few layers to get best results and quicker training times.

We used a 60, 20 and 20% split for training, validation and testing respectively for our model. To further simulate real world conditions we added data augmentation to our model. This allows our model to perform better in difficult scenarios like low lighting, improper camera alignment etc. After experimenting and testing with a lot of different kinds of augmentations we found the following augmentations gave us best results were random zoom that generates extra images that are zoomed in randomly upto 20%. In the same way we added random crops upto 20% and also random brightness for varied conditions. Augmentations effectively increase the size of our dataset and also makes sure the model works better in unknown conditions.

Table 2 Comparison of different models

S. No.	Model name	CPU usage (%)	FPS	Accuracy (%)
1	Inception v2	90	1	98
2	VGG-16	93	1	96
3	Resnet-50	91	2	97
4	**MobileNet v2**	**66**	**6**	**93**

We fine-tuned the MobileNet v2 model on our custom dataset on a computer with a Nvidia GTX1070TI with Cuda acceleration. Our model was trained for roughly 6500 steps with a batch size of 32 which took 4hrs to train and we stopped when we had a validation accuracy of 95.6%.

5 Results and Discussion

In this section we will discuss the results we have received after implementing the system. Our model received 93.6% accuracy on our test dataset and along with training MobileNets we also trained our model on various other architectures such as Inception v2, ResNet-50 and VGG-16 [11] model on the same dataset in order to compare the performance we receive on the Raspberry pi. The results given below are an average of all the 5 runs done on the same Raspberry pi using the above mentioned architectures. Thus after looking at the results we can come to a conclusion that the MobileNet v2 is the most ideal neural net model for our use case. We believe that MobileNet v2 was the best model for our application as we want to have the system to be almost real time which will enable it to prevent accidents. As MobileNet v2 requires the least amount of CPU usage some processing ability of the limited compute on a Raspberry pi cpu can also be used for other tasks such as calculating speed and sending data to web server. We further tested our system in real world conditions and found satisfactory results (Table 2).

6 Conclusion and Future Scope

On successful implementation the system will provide a robust and efficient method to monitor driver activities and thus prevent accidents that occur due to distracted driving, overuse of mobile phones while driving, texting on phone, drowsiness, sleeping etc. When such a system is in place it will enforce the drivers to be more careful and drive responsibly which will prevent loss of lives and will promote a safer driving experience for other drivers on the road.

Once applied over a large number of vehicles the system can also be used to create a network of vehicles to share important information. In the future this network will

be able to collect huge amounts of data and this data can be used to plan routes better. Moreover since we have deployed a hardware platform more and more features can be added in due time. Features such as facial recognition for authentication and various kinds of analysis can be done using the data of our platform. The algorithm in the proposed system relies on a neural network to detect driver's activity which performs well but an object detection approach can be used to detect specific distracting objects which will theoretically perform even better than standard neural net approaches.

References

1. LeCun Y, Bottou L, Bengio Y, Haffner P Gradient based learning applied to document recognition
2. He K, Zhang X, Ren S, Sun J Deep residual learning for image recognition
3. Szegedy C et al (2015) Going deeper with convolutions. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), Boston, MA, pp 1–9
4. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) MobileNets: efficient convolutional neural net
5. Dong Y, Hu Z, Uchimura K, Murayama N (2011) Driver inattention monitoring system for intelligent vehicles: a review. IEEE Trans Intell Transp Syst 12(2):596–614
6. Vicente F, Huang Z, Xiong X, Torre F, Zhang W, Levi D (2015) Driver gaze tracking and eyes off the road detection system. IEEE Trans Intell Trans Syst 16(4):2014–2027
7. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L (2018) MobileNetV2: inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF conference on computer vision and pattern recognition. Salt Lake City, UT, pp 4510–4520
8. Weng C-H, Lai Y-H, Lai S-H (2016)D river drowsiness detection via a hierarchical temporal deep belief network. In: Asian conference on computer vision workshop on driver drowsiness detection from video. Taipei, Taiwan
9. Kaggle Statefarm Dataset https://www.kaggle.com/c/state-farm-distracted-driver-detection
10. Koesdwiady A, Bedawi S, Ou C, Karray F (2017) End-to-end deep learning for driver distraction recognition. In: Image analysis and recognition, 14th international conference, ICIAR 2017, July 2017
11. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations